AMENDMENTS TO THE CLAIMS

The following listing of claims will replace all prior versions and listings of claims in the application.

LISTING OF CLAIMS

1. (Currently Amended) A data processing system for executing a plurality of instructions having a prescribed program order, the data processing system comprising:

a register file including a plurality of registers to store data;

a reorder buffer including N buffer locations of which M buffer locations are bypassable and N-M buffer locations are non-bypassable, wherein N and M are integers and $N > M \geq 0$; and

a plurality of functional units, each functional unit capable of executing instructions regardless of the prescribed program order, and

wherein the reorder buffer temporarily stores data corresponding to the plurality of instructions and, when data of one of the plurality of instructions to be executed by a corresponding one of the plurality of ~~functions units~~ functional units is temporarily stored in one of the M bypassable buffer locations, the detained one of the bypassable M buffer locations is transferred to the corresponding one of the functional units in order to execute the instruction, and wherein the register file also stores data corresponding to retired ones of the plurality of instructions.

2. (Currently Amended) The data processing system of claim 1, wherein the reorder buffer ~~includes~~ is a first-in first-out (FIFO) buffer having the N buffer locations, and

wherein, as one of the plurality of instructions is retired, the data in the nth one of the N buffer locations is transferred to the register file and the detained ~~the~~ nth - 1 one of the N buffer locations is shifted to the nth one of the N buffer locations, and

wherein, when data of one of the plurality of instructions to be executed by a corresponding one of the plurality of functional units is temporarily stored in one of the N-M non-bypassable buffer locations, the corresponding one of the plurality of functional units waits until the data of one of the plurality of instructions is shifted from the N-M nonbypassable buffer locations to the M ~~non~~-bypassable buffer locations.

3. (Currently Amended) The data processing system of claim 2, wherein the reorder buffer ~~comprising~~ comprises control circuitry to transfer the data in one of the bypassable M buffer locations to one of the functional units and to shift the detained nth – 1 one of the N buffer locations.

4. (Original) The data processing system of claim 1, wherein one of the functional units includes a first arithmetic logic unit (ALU) in combination with a load/store unit sharing one or more components.

5. (Original) The data processing system of claim 4, wherein one of the functional units is a second ALU unit and wherein both the first ALU unit and the second ALU unit can execute an ALU instruction, respectively, in same cycles.

6. (Original) The data processing system of claim 1, wherein the register file includes a plurality of inputs to access data as operands to execute one of the plurality of instructions and includes a plurality of outputs to output the operands for one or more instructions in a same cycle.

7. (Original) The data processing system of claim 1, wherein the register file includes one or more write ports to receive data for storage from the reorder buffer.

8. (Original) The data processing system of claim 7, wherein the reorder buffer includes one or more output ports to send data from one or more retired instructions in the N buffer locations.

9. (Currently Amended) A data processing system for executing a plurality of instructions having a prescribed program order, the data processing system comprising:

a register file means including a plurality of register means to store data;

a buffering means including N buffer locations of which M buffer locations are bypassable and N-M buffer locations are non-bypassable, wherein N and M are integers and $N > M \geq 0$; and

a plurality of functional means, each functional means capable of executing instructions regardless of the prescribed program order, and

wherein the buffering means temporarily stores data corresponding to the plurality of instructions and, when data of one of the plurality of instructions to be executed by a corresponding one of the plurality of functional means is temporarily stored in one of the M bypassable buffer locations, the data in one of the bypassable M buffer locations is transferred to the corresponding one of the functional means in order to execute the instruction, and

wherein the register file means also stores data corresponding to retired ones of the plurality of instructions.

10. (Currently Amended) The data processing system of claim 9, wherein the reorder buffer includes a first-in first-out (FIFO) buffer means having the N buffer locations, and

wherein, as one of the plurality of instructions is retired, the data in the nth one of the N buffer locations is transferred to the register file means and the detain the nth - 1 one of the N buffer locations is shifted to the nth one of the N buffer locations, and

wherein, when data of one of the plurality of instructions to be executed by a corresponding one of the plurality of functional means is temporarily stored in one of the N-M non-bypassable buffer locations, the corresponding one of the plurality of functional means waits until the data of one of the plurality of instructions is shifted from the N-M nonbypassable buffer locations to the M ~~non~~-bypassable buffer locations.

11. (Original) The data processing system of claim 10, wherein the buffer means comprises circuitry means to transfer the data in one of the bypassable M buffer locations to one of the functional means and to shift data in the N buffer locations.

12. (Original) The data processing system of claim 9, wherein one of the functional means includes a first arithmetic logic (LU) means in combination with a load/store means sharing one or more components.

13. (Original) The data processing system of claim 12, wherein one of the functional means is a second ALU means and wherein both the first ALU means and the second ALU means can execute an ALU instruction, respectively, in same cycles.

14. (Original) The data processing system of claim 9, wherein the register file means includes a plurality of inputs to access data as operands to execute one of the plurality of instructions and includes a plurality of outputs to output the operands for one or more instructions in a same cycle.

15. (Original) The data processing system of claim 9, wherein the register file means includes one or more write ports to receive data for storage from the buffer means.

16. (Original) The data processing system of claim 15, wherein buffer means includes one or more output ports to send data from one or more retired instructions in the N buffer locations.

17. (Currently Amended) In a processor for executing a plurality of instructions having a prescribed program order, the processor comprising:

a register file including a plurality of registers to store instruction data;

a reorder buffer including N buffer locations of which M buffer locations are bypassable and N-M buffer locations are non-bypassable, wherein N and M are integers and N > M $\geq$ 0, wherein the reorder buffer temporarily stores data corresponding to the plurality of instructions;

a plurality of execution units, each execution unit capable of executing instructions regardless of the prescribed program order, and

an issue logic to issue data from the register file to one of the N buffer locations and, wherein when data of one of the plurality of instructions to be executed by a corresponding one of the plurality of ~~functions~~ execution units is temporarily stored in one of the M bypassable buffer locations, the issue logic transfers data stored or to be stored in one of the bypassable M buffer locations to the corresponding one of the execution units in order to execute the instruction.

18. (Original) The processor of claim 17, wherein the register file also stores data corresponding to retired ones of the plurality of instructions.

19. (Currently Amended) The processor of claim 18, wherein the reorder buffer includes control circuitry to transfer, as one of the plurality of instructions is retired, the data in the nth one of the N buffer locations to the register file, and to shift the detained ~~the~~ nth - 1 one of the N buffer locations to the nth one of the N buffer locations.

20. (Original) The processor of claim 19, wherein the control circuitry waits when data of one of the plurality of instructions to be executed by a corresponding one of the plurality of execution units is temporarily stored in one of the non-bypassable N-M buffer locations shifts into one of the bypassable M buffer locations before transferring the data in one of the bypassable M buffer locations to the corresponding one of the plurality of execution units.

21. (Original) The processor of claim 17, wherein the control circuitry includes an execution matching circuitry that checks if a tag signal associated with an executed instruction by one of the plurality of execution units matches a tag field in one of the M buffer locations and allows a result of the executed instruction to be stored in a buffer location having a matching tag field.

22. (Original) The processor of claim 17, wherein the control circuitry further includes retire circuitry to retire data for one or more instructions in the reorder buffer to the register file.

23. (Original) The processor of claim 17, wherein the control circuitry further includes data hazard detection circuitry that detects a data hazard condition in data.

24. (Original) The processor of claim 23, wherein the control circuitry further includes bypass circuitry, wherein if a data hazard condition is detected, the bypass circuitry is capable of transferring data in one of the bypassable M buffer locations to the issue logic.

25. (Original) The processor of claim 17, wherein the issue logic issues data for one or more instructions to the reorder buffer.

26. (Original) The processor of claim of claim 17, wherein the processor is at least one of an embedded processor or a microprocessor.

27. (Currently Amended) In a processor for executing a plurality of instructions having a prescribed program order, the processor comprising:

a register file means including a plurality of register means to store data;

a buffering means including N buffer locations of which M buffer locations are bypassable and N-M buffer locations are non-bypassable, wherein N and M are integers and $N > M \geq 0$, wherein the buffering means temporarily stores data corresponding to the plurality of instructions;

a plurality of execution means, each execution means capable of executing instructions regardless of the prescribed program order, and

an issue logic means to issue data from the register file means to one of the N buffer locations and, wherein when data of one of the plurality of instructions to be executed by a corresponding one of the plurality of execution means is temporarily stored in one of the M bypassable buffer locations, the issue logic means transfers data stored or to be stored in one of the bypassable M buffer locations to the corresponding one of the execution means in order to execute the instruction.

28. (Original) The processor of claim 27, wherein the register file means also stores data corresponding to retired ones of the plurality of instructions.

29. (Currently Amended) The processor of claim 28, wherein the buffering means includes control circuitry means to transfer, as one of the plurality of instructions is retired, the data in the nth one of the N buffer locations to the register file, and to shift the detained ~~the~~ nth - 1 one of the N buffer locations to the nth one of the N buffer locations.

30. (Original) The processor of claim 29, wherein the control circuitry means waits when data of one of the plurality of instructions to be executed by a corresponding one of the plurality of execution means is temporarily stored in one of the non-bypassable N-M buffer locations shifts into one of the bypassable M buffer

locations before transferring the data in one of the bypassable M buffer locations to the corresponding one of the plurality of execution means.

31. (Original) The processor of claim 29, wherein the control circuitry means includes an execution matching circuitry means that checks if a tag signal associated with an executed instruction by one of the plurality of execution means matches a tag field in one of the M buffer locations and allows a result of the executed instruction to be stored in a buffer location having a matching tag field.

32. (Original) The processor of claim 27, wherein the control circuitry means further includes retire circuitry means to retire data for one or more instructions in the buffering means to the register file means.

33. (Original) The processor of claim 27, wherein the control circuitry means further includes data hazard detection circuitry means that detects a data hazard condition in data.

34. (Original) The processor of claim 33, wherein the control circuitry means further includes bypass circuitry means, wherein if a data hazard condition is detected, the bypass circuitry means is capable of transferring data in one of the bypassable M buffer locations to the issue logic means.

35. (Original) The processor of claim 27, wherein the issue logic means issues data for one or more instructions to the buffering means.

36. (Original) The processor of claim 27, wherein the processor is at least one of an embedded processor or a microprocessor.

37. (Currently Amended) A reorder buffer for executing instructions out-of-order, comprising:

a first-in-first-out (FIFO) buffer having N buffer locations of which M buffer locations are bypassable and N-M buffer locations are non-bypassable, wherein N and M are integers and $N > M \geq 0$; and

wherein the FIFO buffer temporarily stores data corresponding to a plurality of instructions for execution, and, when data of one of the plurality of instructions to be executed is temporarily stored in one of the bypassable M buffer locations, the data in one of the bypassable M buffer locations is transferred to an execution unit corresponding to the instruction to be executed.

38. (Original) The reorder buffer of claim 37, further comprising:

control circuitry to transfer the data in one of the bypassable M buffer locations to the execution unit corresponding to the instruction to be executed.

39. (Original) The reorder buffer of claim 38, wherein the control circuitry, as one of the plurality of instructions is retired, transfers the data in the nth one of the N

buffer locations to a register file and shifts the data in the nth - 1 one of the N buffer locations to the nth one of the N buffer locations.

40. (Original) The reorder buffer of claim 39, wherein the control circuitry waits for the data to move from one of the non-bypassable N - M buffer locations into one of the bypassable M buffer locations before transferring the data.

41. (Original) The reorder buffer of claim 38, wherein the control circuitry comprises:

a state machine to determine if the reorder buffer is full or empty and to control whether any of the N buffer locations receives data.

42. (Original) The reorder buffer of claim 38, wherein the control circuitry further comprises:

a hazard detection circuitry to detect a hazard condition including a data dependency condition.

43. (Original) The reorder buffer of claim 42, wherein the control circuitry further comprises bypass circuitry to receive hazard condition information from the hazard detection circuitry and to transfer the data in one of the bypassable M buffer locations to an execution unit corresponding to the instruction to be executed.

44. (Original) The reorder buffer of claim 38, wherein the control circuitry further comprises execution match detection circuitry that detects if a tag signal from the execution unit matches a tag field in one of the bypassable M buffer locations, wherein if a match exists the execution match detect circuitry allows the bypass circuitry to transfer the data.

45. (Original) The reorder buffer of claim 38, wherein the control circuitry further comprises an entry mux circuitry to allocate entries in the N buffer locations of the reorder buffer for storing data.

46. (Original) The reorder buffer of claim 38, wherein the control circuitry further comprises retire logic circuitry to retire data for one or more instructions to a register file.

47. (Currently Amended) A reorder buffer for executing instructions out-of-order, comprising:

a first-in-first-out (FIFO) buffer means having N buffer locations of which M buffer locations are bypassable and N-M buffer locations are non-bypassable, wherein N and M are integers and $N > M \geq 0$; and

wherein the FIFO buffer means temporarily stores data corresponding to a plurality of instructions for execution, and, when data of one of the plurality of instructions to be executed is temporarily stored in one of the bypassable M buffer

locations, the data in one of the bypassable M buffer locations is transferred to an execution unit corresponding to the instruction to be executed.

48. (Original) The reorder buffer of claim 47, further comprising:

control circuitry means to transfer the data in one of the bypassable M buffer locations to the execution unit corresponding to the instruction to be executed.

49. (Currently Amended) The reorder buffer of claim 48, wherein the control circuitry means, as one of the plurality of instructions is retired, transfers the data in the nth one of the N buffer locations to a register file and shifts the detained ~~the~~ nth - 1 one of the N buffer locations to the nth one of the N buffer locations.

50. (Original) The reorder buffer of claim 49, wherein the control circuitry means waits for the data to move from one of the non-bypassable N - M buffer locations into one of the bypassable M buffer locations before transferring the data.

51. (Original) The reorder buffer of claim 48, wherein the control circuitry means comprises:

a state machine means to determine if the reorder buffer is full or empty and to control whether any of the N buffer locations receives data.

52. (Original) The reorder buffer of claim 48, wherein the control circuitry means further comprises: a hazard detection circuitry means to detect a hazard condition including a data dependency condition.

53. (Original) The reorder buffer of claim 52, wherein the control circuitry means further comprises bypass circuitry means to receive hazard condition information from the hazard detection circuitry means and to transfer the data in one of the bypassable M buffer locations to an execution unit corresponding to the instruction to be executed.

54. (Original) The reorder buffer of claim 48, wherein the control circuitry further comprises execution match detection circuitry means that detects if a tag signal from the execution unit matches a tag field in one of the bypassable M buffer locations, wherein if a match exists the execution match detect circuitry allows the bypass circuitry to transfer the data.

55. (Original) The reorder buffer of claim 48, wherein the control circuitry means further comprises an entry mux circuitry means to allocate entries in the N buffer locations of the reorder buffer for storing data.

56. (Original) The reorder buffer of claim 48, wherein the control circuitry means further comprises retire logic circuitry means to retire data for one or more instructions to a register file.

57. (Currently Amended) In a reorder buffer having ~~a~~ N buffer locations of which M buffer locations are bypassable and N-M buffer locations are non-bypassable, wherein N and M are integers and N > M $\geq$ 0, a method comprising:

fetching one or more instructions; decoding one or more instructions; checking if a data hazard condition exists from one of the decoded instructions;

if a data hazard condition exists, checking if data is available in one of the bypassable M buffer locations for executing one of the decoded instructions; and

transferring the data if available in one of the bypassable M buffer locations to an execution unit for executing one of the decoded instructions.

58. (Original) The method of claim 57, wherein the data hazard condition is a data dependency condition.